LA-UR-21-22421
Approved for public release; distribution is unlimited.

Title: Overview of the ALDX Common Modeling Framework

Author(s): Hickmann, Kyle Scott

Intended for: Internal presentation to the HE working group

Issued: 2021-03-11

Los Alamos NATIONAL LABORATORY

Overview of the ALDX Common Modeling Framework

Kyle Hickmann, XCP-8

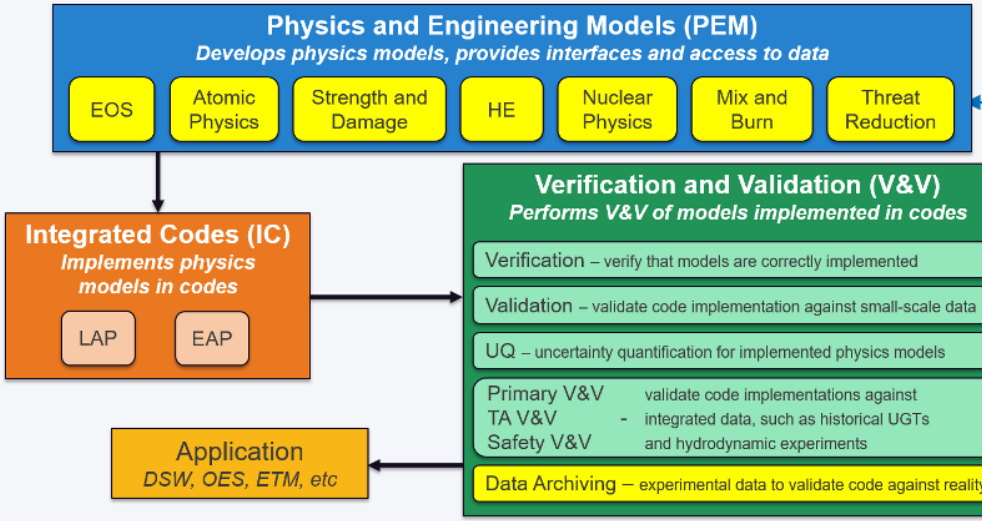
2021/03/10

What are the benefits of using CMF?

- Experiment in CMF is ready for deep VVUQ analysis
  - **Eliminates start-up time** of getting a first simulation ready
  - All VVUQ with a CMF experiment starts from the same place, analysis **results are *comparable***
- A material model in CMF can be used in many experiments by the same researcher
- A researcher using CMF automatically shares their simulation setup and analysis with the division and they leverage the work of all CMF contributors
- **CMF provides a starting place** to set up a simulation
  - Many **material models** are already there
  - Many **meshing examples** are already complete
  - Many **ALE strategies** are available
  - Many **post-processing capabilities** already exist

CMF is intentionally designed to take advantage of the "Data → Model → Validation → Application" workflow pipeline of ASC.

CMF is designed to work in concert with PEM, IC, and V&V to provide a better connection between projects and with the applications.



CMF is working on building interfaces to **PEM data repositories**… this will automate availability of these repos for all CMF users.

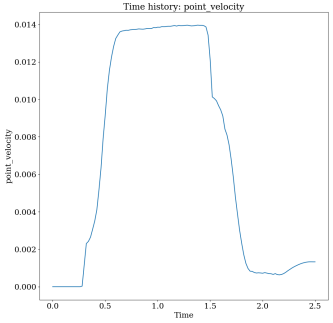
Projects can inherit from each other within CMF to facilitate the flow from small-scale to integrated experiments.
This means that instead of each project creating its own model (as was done in the past), modelers are freed up to **focus on the physics** and provide a higher quality product.

CMF is working on building interfaces to **experimental data repositories**… this will automate availability of these repos for all CMF users.

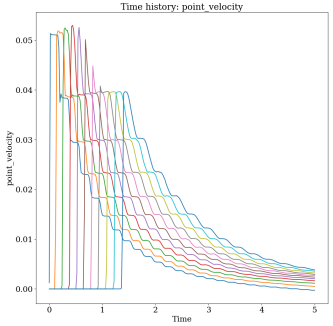
Overview of the Common Modeling Framework

- “What is the CMF?”
  - A large-ish python project within ALDX created to aid design, assessment, and VVUQ.
- “Yeah, but what *is* CMF?”
  - An effort to standardize input deck generation, execution, and analysis while providing a repository for modeling recommendations from PEM, IC, VVUQ, and other projects.
- “If I check out the CMF git repo, what will I see?”
  - A set of python modules that each control the generation of specific input deck sections for one of the ASC codes.
  - A class that organizes sections into a standardized input deck.
  - Sets of defaults for material models and per-experiment model setups.

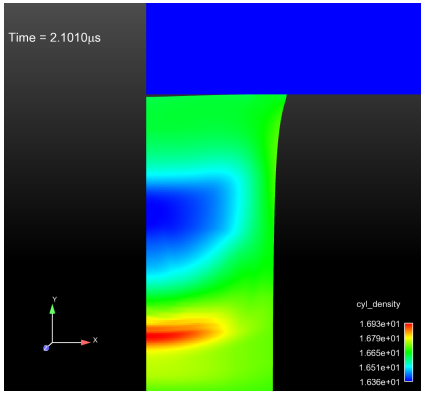
The LAPA simulation suite



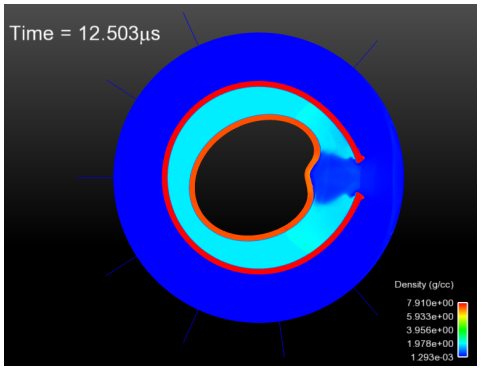
Single layer, windowed flyers in EAPA & LAPA



Embedded gauge flyers in EAPA & LAPA



Taylor anvil in LAPA



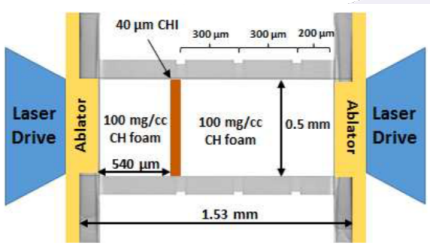
Cyclops in LAPA

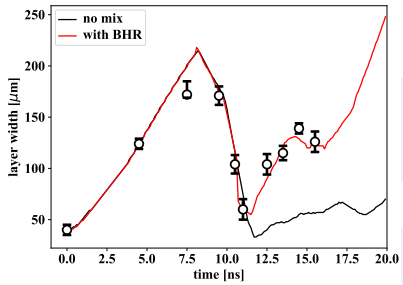
A wide variety of simulations (~70 simulations) have been entered under the LAPA authority

- Inert, HE, and reactive-polymer “flyerplate” experiments
- HE cylinder tests
- Ejecta coupon tests
- Taylor anvil experiments
- “Corner-turning” HE cylinders
- Sushi and Fich experiments
- Cyclops
- Cagliostro

The EAPA simulation suite

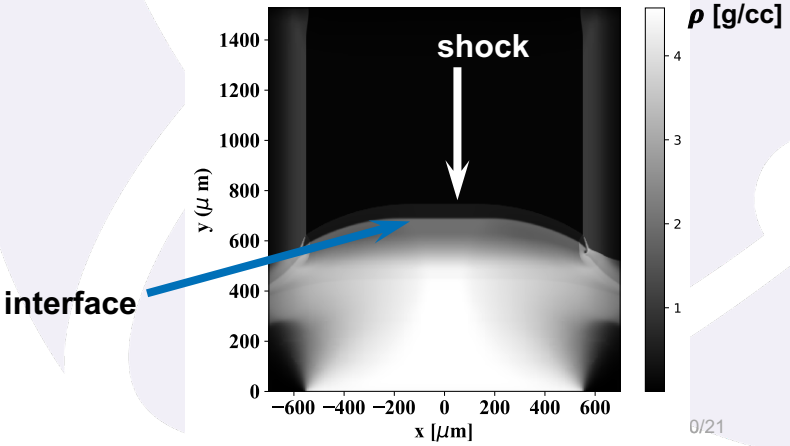
- The EAPA simulation suite is newer (~50 simulations)
  - HE flyerplate experiments
  - HE cylinder tests
  - Poggi shock tube
  - NIF implosion (extremely simplified)
  - Radiation flow test
  - Shaped charge
  - Modcons laser experiments
  - Omega laser Mshock experiments



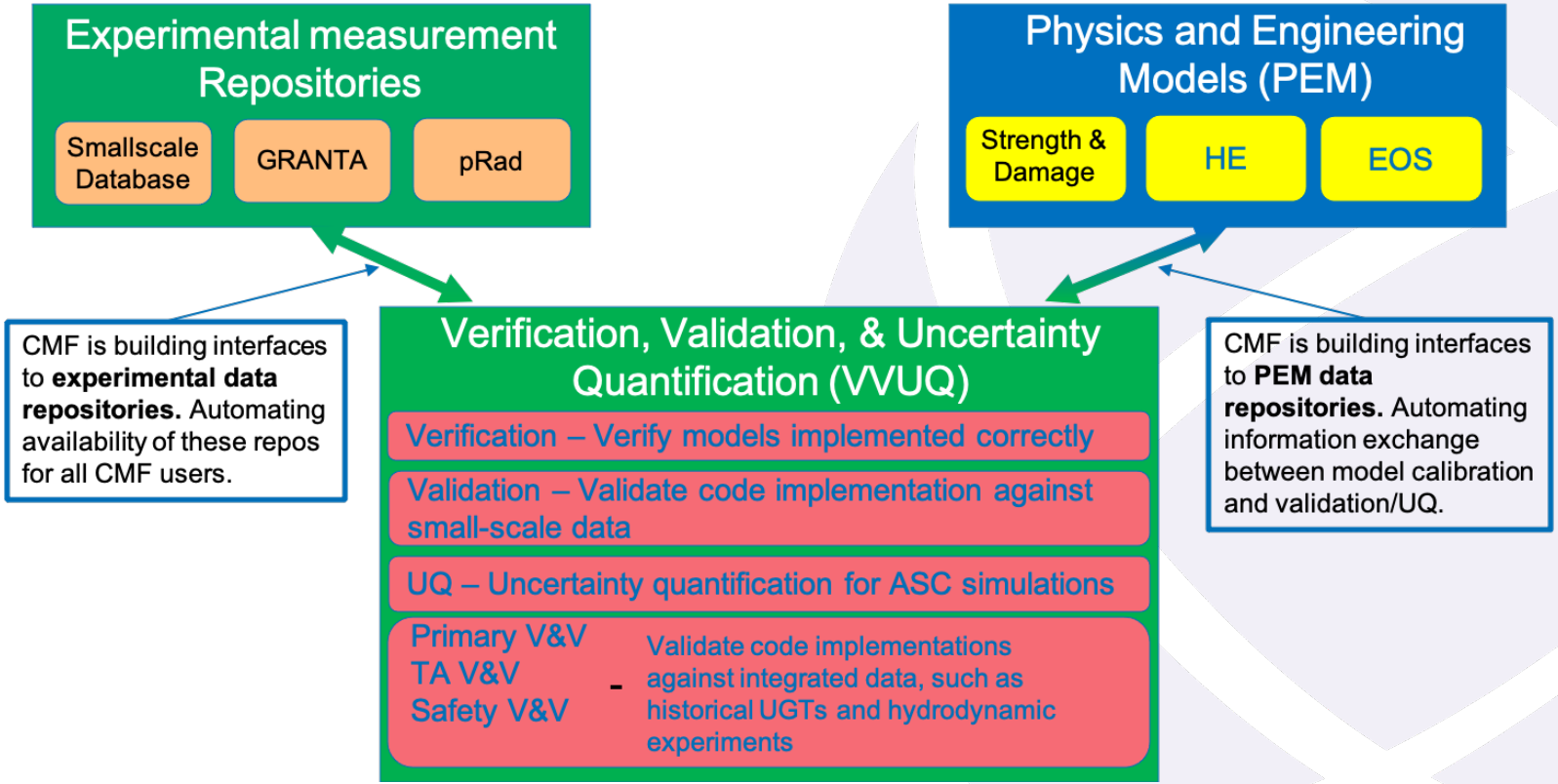


Geometry and results of *Mshock* experiment in EAPA

ModCons density map generated in EAPA



Flow of data within OES & ASC to VVUQ



Shared material models in CMF

- `ModMat` classes store the material property classes
- `SimMat` classes call a code-specific method on material property classes to print blocks of input decks
- Shared material-property classes are under development that contain methods for xRage and Flag
- Allows for easy cross-code VVUQ
- Shared material classes have several CS advantages:
  - Built in type checking
  - Unit-aware attributes
  - Pseudo-database structure of parameter sets

Shared material models database

- For each material multiple parameterizations exist for a single material property
- Selection of a single parameterization based on
  - **Source:** Publication, memo, note specification
  - **Material:** Copper, PBX9501, etc.
  - **Property model:** PTW, Sesame EOS, SURF, etc.
  - **Use:** When to use this parameterization.
- By specifying any 3, a collection of parameterizations is available to use in a `ModMat`

PEM material model interface

- Shared material-property classes can be instantiated by a `YAML` file
- The `YAML` format is simple, human readable, unit aware
- Will allow PEM projects like HE and Strength & Damage to maintain a collection of recommendations that CMF automatically ingests
- This is prototyped for the PEM-HE project and we are working on developing the machinery for PEM-S&D
- Goal is to provide a pipeline from PEM calibration to VVUQ evaluation

Interface with Experimental Data

- Experimental data related to CMF simulations is stored in several places
- Through the `afterPost()` method in per-experiment hooks CMF can compare simulation output to experimental data for VVUQ
- Most experimental data used is currently stored in `/usr/projects/cmf/LargeData`
- CMF wants to stay out of the “data curation” business
- Working on developing automatic interfaces with the SmallScale database
- PEM-HE database of HE data
- When complete, CMF post-processing will automatically pull from these databases